Middlesex University London

Middlesex University Research Repository

An open access repository of

Middlesex University research

http://eprints.mdx.ac.uk

Luo, Linbo, Chai, Cheng, Ma, Jianfeng, Zhou, Suiping ORCID logoORCID: https://orcid.org/0000-0002-9920-266X and Cai, Wentong (2018) ProactiveCrowd: modeling proactive steering behaviours for agent-based crowd simulation. Computer Graphics Forum, 37 (1) . pp. 375-388. ISSN 0167-7055 [Article] (doi:10.1111/cgf.13303)

Final accepted version (with author's formatting)

This version is available at: https://eprints.mdx.ac.uk/23648/

Copyright:

Middlesex University Research Repository makes the University's research available electronically.

Copyright and moral rights to this work are retained by the author and/or other copyright owners unless otherwise stated. The work is supplied on the understanding that any use for commercial gain is strictly forbidden. A copy may be downloaded for personal, non-commercial, research or study without prior permission and without charge.

Works, including theses and research projects, may not be reproduced in any format or medium, or extensive quotations taken from them, or their content changed in any way, without first obtaining permission in writing from the copyright holder(s). They may not be sold or exploited commercially in any format or medium without the prior written permission of the copyright holder(s).

Full bibliographic details must be given when referring to, or quoting from full items including the author's name, the title of the work, publication details where relevant (place, publisher, date), pagination, and for theses or dissertations the awarding institution, the degree type awarded, and the date of the award.

If you believe that any material held in the repository infringes copyright law, please contact the Repository Team at Middlesex University via the following email address:

eprints@mdx.ac.uk

The item will be removed from the repository while any claim is being investigated.

See also repository copyright: re-use policy: http://eprints.mdx.ac.uk/policies.html#copy

ProactiveCrowd: Modeling Proactive Steering Behaviors for Agent-based Crowd Simulation

Journal:	*Computer Graphics Forum*
Manuscript ID	Draft
Wiley - Manuscript type:	Original Article
Date Submitted by the Author:	n/a
Complete List of Authors:	Luo, Linbo; Xidian University, School of Cyber Engineering Chai, Cheng; Xidian University, School of Cyber Engineering Ma, Jianfeng; Xidian University, School of Cyber Engineering Zhou, Suiping; Middlesex University, School of Science and Technology Cai, Wentong; Nanyang Technological University, School of Computer Science and Engineering
Keywords:	Behavioral Animation < Animation, Human Simulation < Animation
Abstract:	How to realistically model an agent's steering behavior is a critical issue in agent-based crowd simulation. In this work, we investigate some proactive steering strategies for agents to minimize potential collisions. To this end, a behavior-based modeling framework is first introduced to model the process of how humans select and execute a proactive steering strategies in crowded situations and execute the corresponding behavior accordingly. We then propose behavior models for two inter-related proactive steering behaviors, namely gap seeking and following. These behaviors can be frequently observed in real-life scenarios, and they can easily affect overall crowd dynamics. We validate our work by evaluating the simulation results of our model with the real-world data and comparing the performance of our model with that of another state-of-the-art crowd model. The results show that the performance of our model is better or at least comparable to the compared model in terms of the realism at both individual and crowd level.

SCHOLARONE™
Manuscripts

Volume xx (200y), Number z, pp. 1–12

ProactiveCrowd: Modeling Proactive Steering Behaviors for Agent-based Crowd Simulation

Linbo Luo[1], Cheng Chai[1], Jianfeng Ma[1], Suiping Zhou[2] and Wentong Cai[3]

[1]School of Cyber Engineering, Xidian University, China
[2]School of Science and Technology, Middlesex University, United Kingdom
[3]School of Computer Science and Engineering, Nanyang Technological University, Singapore

For Review Only

Abstract

How to realistically model an agent's steering behavior is a critical issue in agent-based crowd simulation. In this work, we investigate some proactive steering strategies for agents to minimize potential collisions. To this end, a behavior-based modeling framework is first introduced to model the process of how humans select and execute a proactive steering strategies in crowded situations and execute the corresponding behavior accordingly. We then propose behavior models for two inter-related proactive steering behaviors, namely gap seeking and following. These behaviors can be frequently observed in real-life scenarios, and they can easily affect overall crowd dynamics. We validate our work by evaluating the simulation results of our model with the real-world data and comparing the performance of our model with that of another state-of-the-art crowd model. The results show that the performance of our model is better or at least comparable to the compared model in terms of the realism at both individual and crowd level.

Categories and Subject Descriptors (according to ACM CCS): I.3.7 [Computer Graphics]: 3D Graphics and Realism—Animation, I.6.5 [Simulation and Modeling]: Model Development—Modeling methodologies

1. Introduction

Human crowd is an intriguing social phenomena in nature. Crowd modeling and simulation [ZCC*10] has become an important tool to analyze the impact of crowd behavior in a wide range of applications including safety planning, architecture design, virtual reality, military training, etc [ST05, Tha07, PSAB08, TFB*11, ZCLL14]. Among different approaches to crowd simulation, agent-based crowd modeling has emerged as the most popular and powerful one, due to its capability to model heterogeneous individual behaviors. In agent-based crowd simulation, the overall crowd dynamics is determined by the behaviors of individual agents and their interactions.

To model an agent's behavior in a crowd, an important issue is how to model the agent's steering ability to avoid possible collisions with nearby agents and other dynamic obstacles. Most existing work tackles this problem with a reactive approach, that is, an agent examines oncoming collisions and responsively derives a suitable velocity that can avoid these collisions. The derivation of such suitable velocity could be based on different methods, such as synthetic vision [OPOD10] and velocity obstacles [VDBGLM11]. Although these models are effective in producing smooth locomotion of agents, they mainly deal with the reactive response to immediate collisions. In fact, human's efforts to minimize potential collisions also include some navigation strategies at a higher level. For instance, a person may perform a detour from original planned path and proactively seek to an empty space in a crowd (see Figure 1) as she/he perceives such redirection may result in less efforts to avoid future collisions. We refer to such kind of behaviors as proactive steering behaviors. It is essential to incorporate such behaviors to improve the realism of crowd simulation.

Research on proactive steering behavior modeling has emerged as a promising direction in crowd simulation. To realize proactive steering behavior, existing work [PPD07, GNCL14, BP15] usually adopts a prediction-based approach, which infers proactive steering motions based on the prediction of future states of agents. Such approach has to estimate future interactions among agents which could be complex when the number of agents increases. It may also require some cost functions (e.g., energy consumption function in [BP15]) to determine the optimal velocity of agents. However, in real-life situations, a person may not adopt such optimization techniques (e.g., energy minimization) given the time constraint and complexity of surrounding crowd. We believe that people adopt some simple behaviors such as gap seeking and following to achieve proactive steering. Such behaviors are direct and intuitive strategies that can be often observed from real-life scenarios. Thus, we advocate a behavior-based approach to modeling proactive steering.

In this paper, we present ProactiveCrowd, a behavior-based ap-

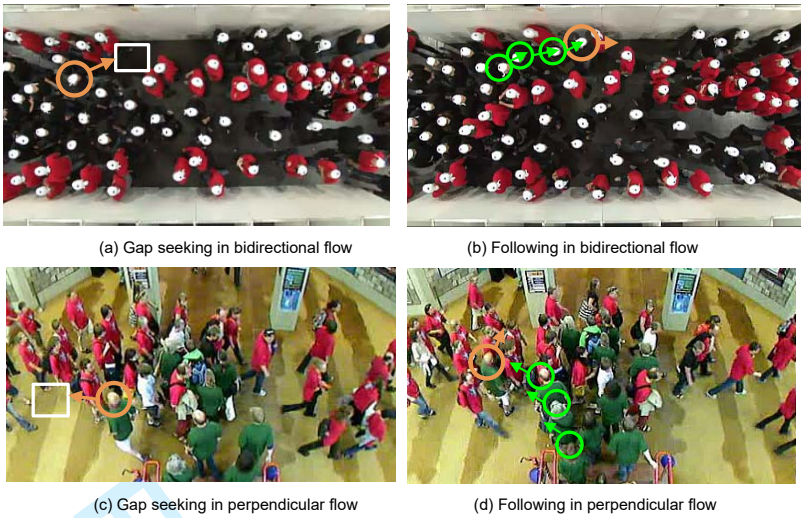

Figure 1: *Gap seeking and following behaviors observed in two real world scenarios. Snapshots for bidirectional flow and perpendicular flow are captured from the crowd experiment videos for the Hermes project [Her] and SMDPC project [SMD] respectively.*

For Review Only

proach for modeling proactive steering behaviors of crowd. We first introduce an overall modeling framework for proactive steering in agent-based crowd simulation. The framework is extensible to incorporate different proactive steering behaviors for complex crowd scenarios. Based on our observations in real-life scenarios, we then propose the behavior models of two types of proactive steering behaviors. The first type of behavior occurs when a gap (i.e., empty space in a crowd flow) is formed due to asynchronized movement of individuals. When a person (see orange circled individual in Figure 1) observes a gap close to her/his walking path, she/he may make a detour to the gap in order to minimize the effort to avoid potential conflicts. We refer this type of behavior as *gap seeking*. Related to *gap seeking* behavior, people (green circled individuals in Figure 1) in the same flow closer to the gap seeker may perform the second type of behavior (i.e., the *following* behavior), as they perceive the path following the gap seeker can help to reduce their efforts for collision avoidance.

The key contributions of this paper are summarized as follows:

- We demonstrate the applicability of mapping intuitive proactive steering strategies of individuals observed from real world scenarios into various behavior models and using a unified framework to drive the behavior selection and execution of these behaviors.
- With the focus on intersecting crowd scenarios, we identify two inter-related proactive steering behaviors, namely *gap seeking* and *following* behavior and propose the detailed behavior models for them. The gap seeking behavior model extends our previous work [LCZM16] by considering the detected gap as a dynamic moving object when the moving direction for gap seeking is computed. Related to gap seeking, the following behavior model not only controls how an agent performs following motion, but also dynmically determines suitable followee (i.e., the individual being followed) from the observed gap seeker and other followers.
- We show the validity of our proposed models by conducting an extensive evaluation with the simulations of two real-world scenarios. The performance of our model is compared with another state-of-the-art crowd model. The simulation results from both models are evaluated against the real world data. The evaluation is performed at both individual level and crowd level using visual inspection, trajectory similarity comparison, fundamental diagrams and progressive distance error. The results show that our model is better or at least comparable to the compared model based on different evaluation metrics.

The remainder of this paper is organized as follows. Section 2 provides an overview of related work. Section 3 describes our modeling framework and the detailed behavior models for the two key proactive steering behaviors, namely *gap seeking* and *following*. In Section 4, we present our case study. Section 5 discusses the conclusion and future work.

2. Related Work

Agent-based crowd modeling and simulation has become an active field in recent years, thanks to its ability to model heterogenous individuals. To model the steering behaviors of a crowd, most agent-based crowd behavior models are built upon two inter-related components: *global navigation* and *local steering*. The global navigation component is usually responsible for generating an agent's path as a series of waypoints within an environment configuration. The local steering component deals with agent's movement from one waypoint to another while avoiding incoming collisions with nearby agents. Different algorithms and models have been proposed to realize these two behavioral components for crowd simulation.

Global navigation is commonly realized using some searching algorithms (e.g., A* algorithm [DP85]) . To support navigation in a complex environment, Shao and Terzopoulos [ST05] proposed a hi-

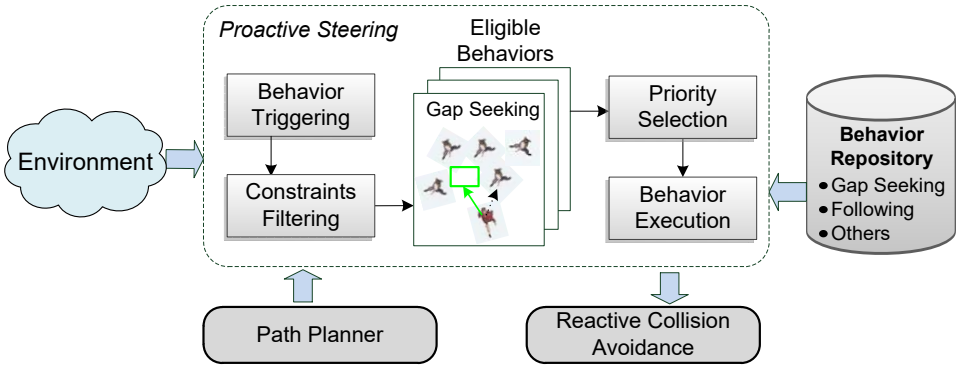

Figure 2: *The overall modeling framework of ProactiveCrowd.*

erarchal environment model to achieve long-range and short-range path planning with A*. Ozcan and Haciomeroglu [OH15] proposed a modified A* algorithm that considers the flow directions of other agents. Other methods include navigation mesh [Sno00] and roadmaps [SGA*07]. Patil et al. [PVdBC*11] adopted a data-driven approach which generates guidance fields from video data to direct an agent's movement.

Extensive work has also been done for modeling the local steering behavior of an individual, which mainly deals with the reactive response to oncoming collisions. The social-force model proposed by Helbing et al. [HFV00] used the physical and socio-psychological forces to model the interactions between agents. Pelechano et al. [PAB07] combined a rule-based system with the force-based model. Karamouzas et al. [KHvBO09] presented a local avoidance method based on collision prediction. Kapadia et al. [KSHF09] proposed an egocentric pedestrian steering framework, which introduces affordance fields in an egocentric manner for local planning and steering. Reciprocal Velocity Obstacles (RVO) model [VDBGLM11] was introduced to generate collision-free locomotion of agent based on velocity space sampling.

More recently, researchers have begun to introduce an additional layer between the global navigation and local steering components. Generally, the purpose of this layer is to enable agents to avoid potential collisions in a more proactive manner. Paris et al. [PPD07] presented a predictive approach which first deduces sets of valid speed and orientation based on the predication of future agent interactions and then selects the best solution using a cost function, considering environment state and an agent's goal. Hu et al. [HZW*10] proposed a framework which generates agent's steering actions based on observed spatial-temporal patterns. Golas et al. [GNCL14] proposed a long-range collision avoidance model which extends the local RVO model by exploring future states of agents with lookahead times. The long-range interactions are estimated and used to add velocity constraints for collision avoidance. Bruneau and Pettré [BP15] introduced a mid-term planning model which also explores future agent interactions within the next n time steps and devises optimal avoidance strategy based on an energy cost function. Best et al. [BNCM14, NBCM15] proposed a DenseSense model which uses a density-dependent filter to correct the preferred velocity from global planner. The filter can be augmented with different local avoidance models (i.e., social force model and RVO) to generate density-dependent crowd behaviors which result in better utilization of free space by the agents.

Different from the existing work for proactive steering behavior modeling, we adopt a behavior-based approach which aims to identify a set of proactive steering strategies that a real person performs in real-life situations. We formalize these strategies as a set of behaviors, such as gap seeking and following, and utilize these behaviors to drive the agents for proactive steering. Compared to the existing work which is mostly based on the prediction of complex future agent interactions, our behavior-based approach provides a more direct and human-like maneuver to capture how individuals proactively minimize potential conflicts in typical crowd scenarios.

In terms of behaviors being modeled, our gap seeking behavior model considers how to better utilize free spaces around an agent's surrounding area, which shares some similarities with the DenseSense model [BNCM14, NBCM15]. However, we use the notion of gap to represent free spaces that are dynamically formed due to the movement of crowds. Thus, instead of evaluating crowd density at different points as in DenseSense, the agents in our model can directly select a suitable gap to move to, which we believe can better imitate the behavior of a real person. In addition, we also model how other agents will follow the gap seeker, which is also commonly observed in real-life. It should be also noted that our gap seeker agents are not equivalent to the leader agents in leader-follower concept in crowd modeling. The leader agents in most of the existing work [PB06, PAB07, ZZC16] are not endowed with proactive steering abilities. They are mostly treated as agents who move based on their planned paths with the determined leading roles.

3. ProactiveCrowd: Behavior-based Proactive Steering Modeling

Figure 2 shows the overall framework for modeling proactive steering in our agent-based crowd simulation. The framework introduces a proactive steering module between the path planning module and the reactive collision avoidance module. The path planner first generates a set of way points for an agent to navigate in a given environment. Along the path to each way point, the proactive steer-

For Review Only

ing module is then invoked to monitor the changing environment, select a suitable behavior (e.g., gap seeking or following) for proactive conflict minimization, and execute the selected behavior from behavior repository. The execution of the selected behavior will dynamically adjust the *desired velocity* which is originally determined by the path planner. The desired velocity adjusted by the proactive steering module is then sent to the reactive collision avoidance module where the *actual velocity* for immediate collision avoidance with nearby agents and obstacles are determined. At each time step, the agent moves according to the actual velocity. It is possible that an agent does not perform any proactive steering behavior in a given situation. In such case, the agent's movement is directly controlled by the underlying reactive collision avoidance module.

Within the proactive steering module, we propose four functional processes to achieve behavior-based proactive steering. The *behavior triggering* process monitors the environment and checks if the condition for triggering a specific behavior is satisfied. For each proactive steering behavior, we define a specific *triggering condition*. For example, for gap seeking behavior, the *triggering condition* is that the agent should detects gaps and its current position is not too close to its goal (i.e., the final destination in the planned path). The detailed specifications of triggering conditions are behavior-dependent, which will be described in the following sub-sections. For each triggered behavior, the *constraint filtering* process is then invoked to further examine the current situation to evaluate the suitability to perform the triggered behavior. For instance, we define a set of *behavioral constraints* (see subsection 3.1.2) to check whether the detected gaps are suitable for gap seeking. If all the detected gaps are filtered out based on the defined constraints, the agent will not perform the gap seeking behavior.

After the *behavior triggering* and *constraint filtering* processes, it is possible that multiple behaviors are eligible to perform . In such cases, we use the *priority selection* process to choose the behavior with the highest priority. In our current implementation, we simply set gap seeking behavior having a higher priority than following behavior (that is, given that an agent detects a suitable gap itself, it prefers to seek to the gap rather than to follow another gap seeker). With more behaviors being incorporated, an advanced priority scheme will be realized in future. The final process is the *behavior execution* process which invokes the motion control algorithm of the selected behavior from the behavior repository and generates the desired velocity accordingly.

In the following two sub-sections, we will describe the modeling of gap seeking and following behaviors respectively, including their triggering conditions, behavior constraints and motion control algorithms.

3.1. Gap Seeking Behavior Model

To trigger gap seeking behavior for an agent, we define the *triggering condition* as an agent detects gaps and its current position is not too close to its goal. Here, we consider how an agent's distance to its current goal affects gap seeking behavior. In reality, a person may be reluctant to perform gap seeking behavior when she/he is approaching her/his goal. Thus, we use a probability measure to decide whether an agent will perform gap seeking behavior given its distance to its goal. Specifically, the probability $C_i(t)$ for agent i at time t to perform gap seeking is determined by:

$$C_i(t) = \lambda \frac{\|\vec{p}_i(t)\|}{S_i}, \tag{1}$$

where $\vec{p}_i(t)$ is the vector from agent i to his goal at time t, S_i is the distance from its initial position to its goal and $\lambda(>1)$ is a bounding parameter. If $C_i(t)$ is greater than 1, it will be set to 1.

Next, we will explain how an agent detects a gap; how to select a suitable gap based on a set of behavioral constraints; and how to control the motion of agent for gap seeking in the following subsections.

3.1.1. Gap Detection

An agent first needs to detect available gaps in its surrounding area. In our framework, agents move in a continuous space. However, it is inefficient to represent a gap in such environment representation. Therefore, we introduce a two-layer virtual world representation for gap detection as shown in Figure 3. The bottom layer is a continues space where agents perform local steering behavior. The top layer is a grid map used for gap detection. In our current implementation, the grid cell size is set to $0.1m \times 0.1m$, and each agent can occupy multiple grid cells.

In the grid map, we use a rectangle shape to represent a gap (see green rectangles in Figure 3). To detect the gaps near an agent, we first define a detection area D (dashed blue rectangle in Figure 3) within which the agent performs gap detection. The detecting agent (i.e., the grey circle in Figure 3) is located at the center of the detection area. For ease of computation, we detect all the gaps surrounding the detecting agent without considering the agent's moving direction. In the *gap selection* step described later on, we will filter out the gaps that are out of agent's vision according to its moving direction. Given the detection area D, we specify the occupancy area O as the area within D which is occupied by other agents. The gap detection operation is thus performed within an exploration area E which is defined as:

$$E = D - O. \tag{2}$$

In the grid map, the exploration area E contains the grid cells in the detection area D which are not occupied by other agents.

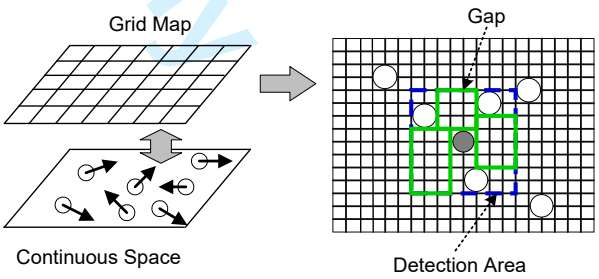

Figure 3: *Layered virtual world representation.*

To detect all available gaps within the exploration area E, a *randomized* algorithm is proposed as shown in Algorithm 1. The algorithm works similar to crystal nucleation and a gap is detected by

For Review Only

Algorithm 1 Gap Detection Algorithm

1: $G \leftarrow \emptyset$ ▷ G is a set of detected gaps
2: $P \leftarrow$ UniformlyPick(E) ▷ find a set of growth seeds uniformly in E
3: **while** P is not empty **do**
4: $p \leftarrow$ PopRandom(P) ▷ randomly select a seed from P
5: $x \leftarrow$ NewRectangle(p.x, p.y) ▷ nucleate at p
6: $F \leftarrow$ {UP, DOWN, LEFT, RIGHT} ▷ allowed growth directions
7: **while** F is not empty **do**
8: $x \leftarrow$Expand(x, F, E) ▷ expand the rectangle until it reaches the boundaries of area E or the cell occupied by other agent
9: Remove the current growth direction from F
10: **end while**
11: Insert(G, x) ▷ Add the detected gap to G
12: **end while**
13: RemoveDuplicate(G) ▷ Remove the duplicate gaps
14: **return** G

expanding a random seed (like a crystal seed) in a greedy manner by consuming nearby uncrystallized cells. A set of seed cells are first uniformly picked from the exploration area E (line 2 in Algorithm 1). Then, a seed cell is randomly selected and expands from one of four growth directions by one cell at a time until it reaches the boundaries of area E or the cell occupied by other agent (lines 4-8 in Algorithm 1). The gap is detected through such rectangle expansion and it is added to the set of all detected gaps G in each iteration (line 10 in Algorithm 1). Due to randomness, the detecting process may yield duplicate gaps from different seed cells. Therefore, the algorithm removes duplicate gaps in the end (line 12 in Algorithm 1).

3.1.2. Gap Selection

Given all the detected gaps in G, an agent needs to select the most desirable gap. In case that G is empty (i.e., no detected gap), the agent will not proceed to gap selection. To select the most desirable gap, we propose four behavioral constraints with given priorities. The behavioral constraint with a higher priority is examined first and if the detected gap does not satisfy the constraint, the gap is discarded without checking the lower priority constraints. This can help to save the computational cost of our model. If no gap is selected after examining all behavioral constraints (i.e., *constraints filtering* process in our framework), the gap seeking behavior will not be performed. The proposed four constraints with the priorities from high to low are described as follows.

Constraint G.1: *A gap should be within the vision of the detecting agent.* As mentioned previously, all surrounding gaps are detected in the gap detection step without considering agent's moving direction. In reality, a person's vision is usually limited to a fan-shape region centered along the person's moving direction as shown in Figure 4. Thus, we need to filter out the gaps that are out of the vision of the detecting agent.

We check whether a gap is within the agent's vision using the following two rules:

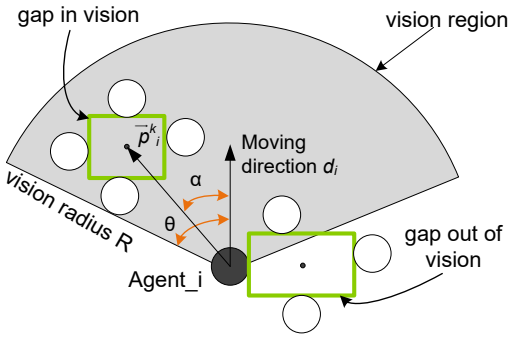

Figure 4: *Selecting gap(s) based on agent's vision.*

$$\|\vec{p}_i^k\| \leq R, \tag{3}$$

$$angle(\vec{d}_i, \vec{p}_i^k) \leq \theta, \tag{4}$$

where $\vec{p}_i^k$ is the vector pointing from agent i's current location to the center of the kth detected gap, $\|\vec{p}_i^k\|$ computes the magnitude of $\vec{p}_i^k$, R is the vision radius of an agent, $\vec{d}_i$ is the vector denoting agent i's moving direction, *angle()* computes the angle between two vectors in 2D space and θ is half of the vision angle of an agent.

Constraint G.2: *A gap should be big enough for the detecting agent to move in.* In our model, if the detected gap is too small, the detecting agent will not consider it as a desirable gap. Since we use rectangle to represent a gap, the following rule is used to determine whether the size of gap is appropriate:

$$\min\{w_i^k, l_i^k\} \geq 2r_i, \tag{5}$$

where w_i^k is the width of the kth gap detected by agent i, l_i^k is the length of the kth gap detected by agent i and r_i is the radius of agent i.

Constraint G.3: *A gap should not deviate too much from the detecting agent's intended moving direction towards final destination.* In our model, we assume that an agent will not choose a gap whose direction from the agent's current position deviates too much from agent's direction towards its destination (i.e., the final goal agent aims to reach). This is because an agent may take more effort to reach its destination if seeking to such gaps. To avoid such gaps, the following rule is used:

$$angle(\vec{u}_i, \vec{p}_i^k) \leq \varphi, \tag{6}$$

where $\vec{u}_i$ is the vector pointing from agent i's current location to its destination and φ is a user-defined threshold angle.

Constraint G.4: *The agent is the closest one to the selected gap among all gap seekers to the same gap*. This situation occurs when multiple gap seekers choose the same gap at the same time. If an agent is not the nearest one to the gap, it will perform the following behavior rather than the gap seeking behavior. By doing this, only the one that is closest to the gap will perform gap seeking and others will follow.

For Review Only

Given that all the above four constraints are satisfied, it is still possible that multiple detected gaps are selected. In such cases, the final selection of the most desirable gap is performed according to the following rule:

$$k^* = \arg\min(angle(\vec{u}_i, \vec{p}_i^k)), \tag{7}$$

This means we select the k^*th gap whose direction $\vec{p}_i^k$ has the minimum angle with agent's direction towards its destination. Here, we assume that an agent selects the gap whose direction is most consistent with the direction towards the agent's destination.

3.1.3. Gap Seeking

In the last step, an agent needs to update the desired velocity $\vec{v}_i^{\,\mathrm{d}}$ so as to move to the selected gap. During gap seeking process, a gap itself also moves due to the movements of its bounding agents. Thus, we consider a gap as a dynamic moving object as shown in Figure 5. To take into account of the dynamic nature of the selected gap, the direction of $\vec{v}_i^{\,\mathrm{d}}$ (i.e., the direction of $\vec{p}_i^{\,t+T^s}$ in Figure 5) is determined with the following steps.

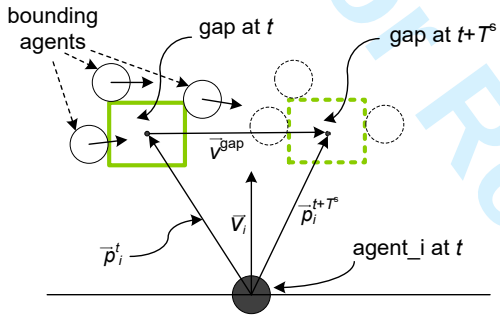

Figure 5: *Modeling gap as a dynamic moving object.*

Step 1: Estimate the time an agent will take to perform gap seeking behavior as follows.

$$T^{\mathrm{s}} = \frac{\|\vec{p}_i^{\,t}\|}{\|\vec{v}_i^{\,\mathrm{d}}\|}, \tag{8}$$

where T^{s} is the estimated time an agent needs to perform gap seeking behavior, $\|\vec{p}_i^{\,t}\|$ is the magnitude of the vector pointing from agent i's current location to the center of the selected gap at the start time t of gap seeking process, $\|\vec{v}_i^{\,\mathrm{d}}\|$ is the desired speed of the agent determined based on Equation 12. It should be noted that T^{s} only represents a rough estimation of time required to perform gap seeking based on the agent's current observations. We believe a real person may adopt a similar way for estimating the gap seeking time, given the time limitation in real-life scenarios.

Step 2: Estimate the moving speed of the gap based on the average speed of its bounding agents as follows.

$$\vec{v}^{\,\mathrm{gap}} = \frac{1}{n}\sum_{j=1}^{n} \vec{v}_j^{\,\mathrm{gap}}, \tag{9}$$

where $\vec{v}^{\,\mathrm{gap}}$ is the velocity of the moving gap, $\vec{v}_j^{\,\mathrm{gap}}$ is the velocity of bounding agents (see Figure 5) of the gap and n is the total number of bounding agents.

Step 3: Determine the position of gap after T^s as follows.

$$\vec{x}^{\,\mathrm{gap}}(t+T^s) = \vec{x}^{\,\mathrm{gap}}(t) + \vec{v}^{\,\mathrm{gap}} T^{\mathrm{s}}, \tag{10}$$

where $\vec{x}^{\,\mathrm{gap}}(t)$ is the position of the gap at time t.

Step 4: Determine the moving direction of agent for gap seeking as follows.

$$\vec{e}_i^{\,d} = \frac{\vec{x}^{\,\mathrm{gap}}(t+T^s) - \vec{x}_i(t)}{\|\vec{x}^{\,\mathrm{gap}}(t+T^s) - \vec{x}_i(t)\|}, \tag{11}$$

where $\vec{e}_i^{\,d}$ is the desired moving direction of the gap seeker agent i (i.e., the unit vector with the direction of $\vec{p}_i^{\,t+T^s}$ in Figure 5), $\vec{x}_i(t)$ is the agent i''s position at the time t. It should be noted that the time T^{s} is only an estimated time based on the current information (i.e., the current location of gap at time t). This time represents an agent's intuition on the time it needs to perform gap seeking and may not correspond to the actual time for gap seeking. Based on this estimated time, the projected position of the dynamic gap after T^{s} is determined. The velocity of the dynamic gap is set as the average velocity of its bounding agents (i.e., the agents who form this gap). The desired moving direction of the agent is thus set from agent's current position towards the projected position of the gap after T^{s}.

The magnitude of $\vec{v}_i^{\,\mathrm{d}}$ (i.e., the speed of gap seeker agent) is set according to the following rule:

$$\|\vec{v}_i^{\,\mathrm{d}}\| = \frac{V_{\max}}{1+\exp[-\beta(s-\alpha s_{\min})]}, \tag{12}$$

where $V_{\max}$ is the maximum speed of an agent, s is the area of the selected gap, $s_{\min}$ is the area of the smallest gap we can select (i.e., $s_{\min} = 4 * r_i^2$ in our model), and $\alpha \in (0,1]$ and $\beta > 0$ are bounding coefficients. According to Equation 12, the speed of the agent increases when the area of the gap increases. Here, we consider a large gap attracts an agent more than a small one (i.e., having a higher chance to successfully seek to the gap). Thus, the agent may prefer to seek with a higher speed. The highest speed an agent can take is $V_{\max}$ and the lowest speed is taken when the area of the gap is smallest (i.e. $s_{\min}$). For example, $\|\vec{v}_i^{\,\mathrm{d}}\| = V_{\max}/2$ when $s = s_{\min}$ and both α and β are set to 1.

While moving according to $\vec{v}_i^{\,\mathrm{d}}$, the agent completes the gap seeking behavior under two conditions. The first condition is when a seeking time T^{s} is expired. The second condition is when the agent has reached the central point of the selected gap. Until it completes the gap seeking behavior, the agent will then perform another cycle of proactive steering process.

3.2. Following Behavior Model

While there are existing efforts [LJK*12, HLPW14, FETA16] to model the following behavior of pedestrians in a crowd, most of them focus on how to simulate the motion of a follower when the followed pedestrian is pre-defined. In our model, a follower agent has to dynamically choose a followee (i.e., the followed pedestrian) among the detected gap seeker and other followers in the same crowd flow. Thus, apart from modeling the motion of follower, we also model how to determine the followee of a follower and how long the following behavior is performed.

For Review Only

The *triggering condition* for the following behavior is that an agent detects a gap seeker or followers of the gap seeker in the same crowd flow. Once the *triggering condition* is met, the next task is to determine whether there is a suitable followee it can follow, which is determined by the following behavioral constraints.

Constraint F.1: *The followee's moving direction must not deviate much from follower's own intended moving direction*:

$$angle(\vec{v}_i^{\,\mathrm{d}}, \vec{v}_{\mathrm{e}}^{\,\mathrm{d}}) \leq \rho, \tag{13}$$

where $\vec{v}_i^{\,\mathrm{d}}$ is the follower agent i's desired velocity, $\vec{v}_{\mathrm{e}}^{\,\mathrm{d}}$ is its followee's desired velocity and ρ is a user-defined deviation angle.

Constraint F.2: *The followee has not been followed by another agent*. Given the space constraint in high-density crowd, we assume that the following behavior is performed as a chain with one agent followed by another one. This constraint is thus to ensure that an agent can only be followed by another agent.

If there is no suitable followee according to the two constraints, the following behavior will not be performed. Otherwise, an agent i probabilistically chooses a followee using a multinomial logit-based model as:

$$P_j = exp(-\tau d_{ij}) / \sum_{k}^{K} exp(-\tau d_{ik}), \tag{14}$$

where P_j is the probability of choosing agent j as followee, d_{ij} is the distance between agent i and j, K is the number of potential followee agents according to the two aforementioned constraints and $\tau(>0)$ is a control parameter. According to Equation 14, a potential followee agent closer to the follower agent has higher probability to be selected as the followee.

Once an agent has selected a followee to follow, it will perform the following behavior until a following time has expired or the followee is out of vision. The following time is determined as:

$$T_j^{\mathrm{f}} = T_{j-1}^{\mathrm{f}} - \Delta t_m = T^{\mathrm{s}} - \sum_{m=1}^{j} \Delta t_m, \tag{15}$$

where T_j^{f} is the following time of the jth follower, T_{j-1}^{f} is the following time of the $(j-1)$th follower (i.e., the followee of jth follower), Δt_m is the time difference between the starting times of the jth and the $(j-1)$th followers to perform following behavior, and T^{s} is the gap seeking time of the gap seeker (i.e., followee of the 1st follower). Given a sequence of gap seeker and its followers, the following time of the follower decreases by Δt_m time. By doing so, we can avoid the following behavior to be propagated indefinitely in the crowd. In real-life, we can observe that only people who are close to a gap seeker are more likely to perform the following behavior. Note that once T_j^{f} becomes less or equal to zero, the jth follower and subsequent agents will not perform the following behavior any more.

During the following process, an follower agent needs to update the desired velocity $\vec{v}_i^{\,\mathrm{d}}$ so as to follow the selected followee. We use a weighted vector summation similar to [HLPW14] to determine the moving direction of a follower agent as follows:

$$\vec{e}_i^{\,\mathrm{d}} = \eta \vec{e}_j + (1-\eta)\vec{n}_{ij}, \tag{16}$$

where $\vec{e}_i^{\,\mathrm{d}}$ is the desired moving direction of follower agent i, $\vec{e}_j$ is the moving direction of the selected followee agent j, $\vec{n}_{ij}$ is the unit vector pointing from agent i to agent j and $\eta = exp(-\kappa * d_{ij})$ is a coefficient which depends on the distance between the follower and the followee (i.e., d_{ij}) with a bounding fraction $\kappa > 0$.

To determine the magnitude of $\vec{v}_i^{\,\mathrm{d}}$, we adopt a velocity-based distance acceleration model which maintains a distance between the follower and the followee according to follower's current speed. The acceleration and the desired speed of a follower agent i are determined as:

$$a_i = \omega(d_{ij} - \xi - \psi \vec{v}_i \cdot \vec{e}_i^{\,\mathrm{d}}), \tag{17}$$

$$\|\vec{v}_i^{\,\mathrm{d}}\| = \vec{v}_i \cdot \vec{e}_i^{\,\mathrm{d}} + a_i \Delta t, \tag{18}$$

where $\vec{v}_i$ is the current velocity of the follower i, Δt is the simulation time step, and ω, ξ, ψ are control parameters that need to determined according to the specific scenario.

4. Case Study

In this section, we conduct a case study by applying our proposed ProactiveCrowd to simulate two real world scenarios involving intersecting crowd flows (i.e., perpendicular flows and bi-directional flows). To evaluate the effectiveness of ProactiveCrowd, we compare it with a state-of-the-art crowd model DenseSense [BNCM14], which introduces a density-dependent filter for proactive crowd steering. The simulation outputs generated by our model (i.e., ProactiveCrowd) and DenseSense are compared against the real world data extracted from the real-world scenarios. To comprehensively evaluate the performance of our model, the simulation results are evaluated at both individual-behavior level and crowd level. In this section, we first describes the scenario being modeled and the simulation settings. Then, we present our experiments at two levels and the simulation results.

4.1. Scenarios Description

(a) Perpendicular crowd scenario (b) Bi-directional crowd scenario

Figure 6: *Sample frames of perpendicular and bi-directional crowd scenarios. Snapshots are obtained from [SMD] and [Her] respectively.*

We apply ProactiveCrowd to simulate two typical real-world crowd scenarios. The first one is a perpendicular crowd scenario where two crowd flows move with an intersecting angle of around 90 degree (see Figure 6 (a)). The second one is a bi-directional

For Review Only

crowd scenario where two crowd flows move in opposite directions. The first scenario was performed by Plaue et al. at Technische Universität Berlin during the Long Night of the Sciences 2010 in Berlin [PCBS11] and the second scenario was performed be Zhang et al. in hall 2 of the fairground at Dusseldorf (Germany) [ZKSS12].

In the first scenario, one group of 54 individuals moves from left to right on a plain ground. Another group of 46 individuals moves from bottom to top starting at a 1.81m wide staircase. The region where the two crowd flows intersect is about 3m×3m. From the recorded videos of the scenario, we can observe the proactive steering behaviors including both the gap seeking behavior and the following behavior among the crowd. We obtain the individuals' trajectories data from [SMD]. The trajectory data are captured at 15 fps.

In the second scenario, Zhang et al. [ZKSS12] have performed several experiments under different crowd and environment settings. In this work, we choose a dataset where 141 individuals move from left to right and 163 individuals move from right to left in a the corridor with width of 3.6m and length of 8m. The destinations of individuals were chosen based on a number given to them in advance (odd-numbered individuals move to the left at the end of the corridor, even-numbered individuals move to the right at the end of the corridor). For the recorded video, we can also observe the gap seeking behavior and following behavior among the crowd. We obtain the individuals' trajectories data from [Her]. The trajectory data are captured at 16fps.

4.2. Simulation Settings

Our agent-based crowd simulation is implemented with Repast Simphony[†]. 3D visualization is realized in Unity3D[‡]. As the environment of the simulated scenario is relatively simple, we do not use any path planning algorithm to generate a path for global navigation. The target location of an agent is directly set to its corresponding individual's location at the last frame in the trajectories data. The initial location of an agent is set based on the corresponding individual's first position in the trajectories data. The initial speed of an agent is set to be the average speed of the corresponding individual in the first three recorded frames. The local steering behavior for avoiding oncoming collisions is implemented using the social-force model [HFV00]. The DenseSense model is also an augmented model which can choose different local collision avoidance algorithms for local steering. For fair comparison, the DenseSense model also operates over the social-force model.

Using the recorded data of the two scenarios being studied, we have calibrated our model and the DenseSense model using evolutionary algorithm [JHS07] with 6-fold cross validation. For the gap seeking behavior model, the values of calibrated parameters are as follows. The detection area D is 3m×3m. The vision radius and angle of an agent is set to 2.5m and 120 degree respectively. α and β values in Equation 12 are set to be 0.5 and 0.75 respectively. $V_{\max}$ of an agent is $1.34m/s$ and the radius of agent is 0.25m. For the following behavior model, the calibrated parameters are as follows. The deviation angle ρ is 120 degree. The control parameter τ in Equation 14 is set to 0.65. The bounding fraction κ in Equation 16 is 0.26. The control parameters ω, ξ, ψ in the follower's acceleration Equation 17 are 1.2, 0.35, and 0.65 respectively. For the DenseSense model, the stride factor and stride buffer for determining the natural walking speed are set to be 3.0 and 0.2 respectively. The σ parameter in the Gaussian density function for determining density of neighbouring agents is set to 2.2. The simulation time step is set to be the same as the frame rate in the trajectories data (i.e., 0.067s for perpendicular crowd scenario and 0.0625s for bi-directional crowd scenario).

4.3. Experimental Results and Discussions

4.3.1. Individual Level Evaluation

To evaluate how our model can replicate the individual-level proactive steering behaviors that are observed in real crowds, we conducted both the visual inspection through 3D visualization and qualitative evaluation using a trajectory similarity metric.

(a) *Visual inspection of individual behaviors*

To visually observe the individual behaviors in the scenario, we have visualized simulation results produced by our model, ProactiveCrowd, and the DenseSense model in Unity3D and compared them with the corresponding real world crowd data. Figure 7 shows the snapshots of the 3D visualizations for the two scenarios respectively. For more details, a demo video can be found in the link[§]. To make it easy to observe the agents performing proactive steering behavior in our model, we color-code the gap seeker agents and follower agents as orange-shirt and green-shirt agents respectively. The corresponding agents in the DenseSense model are populated using the same color-code and these individuals are circled in the real world crowd data. From Figure 7, it can be observed that our model is able to replicate both gap seeking and following behaviors of agents that can be observed in real world data. However, the corresponding agents in the DenseSense model do not exhibit such gap seeker-followers patterns under the same situation. Furthermore, it can also be observed that the gap seeker and follower agents can affect the lane formations especially in the bi-directional crowd scenario. As such, the crowd dynamics generated by our model also have a closer match to that in real world data.

(b) *Trajectory similarity evaluation*

To quantitatively evaluate the performance of ProactiveCrowd, we compare the trajectory similarity between our agents and the corresponding human data. We adopt the Longest Common Subsequence (LCSS) metric which has been proven to be very robust to noise for trajectory similarity comparison [VKG02]. The LCSS similarity metric is calculated as follows:

$$S_{LCSS}(Tr^{\text{real}}, Tr^{\text{sim}}) = \frac{LCSS(Tr^{\text{real}}, Tr^{\text{sim}})}{\min(m,n)} \qquad (19)$$

[†] Repast: http://repast.sourceforge.net/repast_simphony.php

[‡] Unity3D: http://unity3d.com/

[§] Demo video: https://youtu.be/kcJ_B1eOS5Y

For Review Only

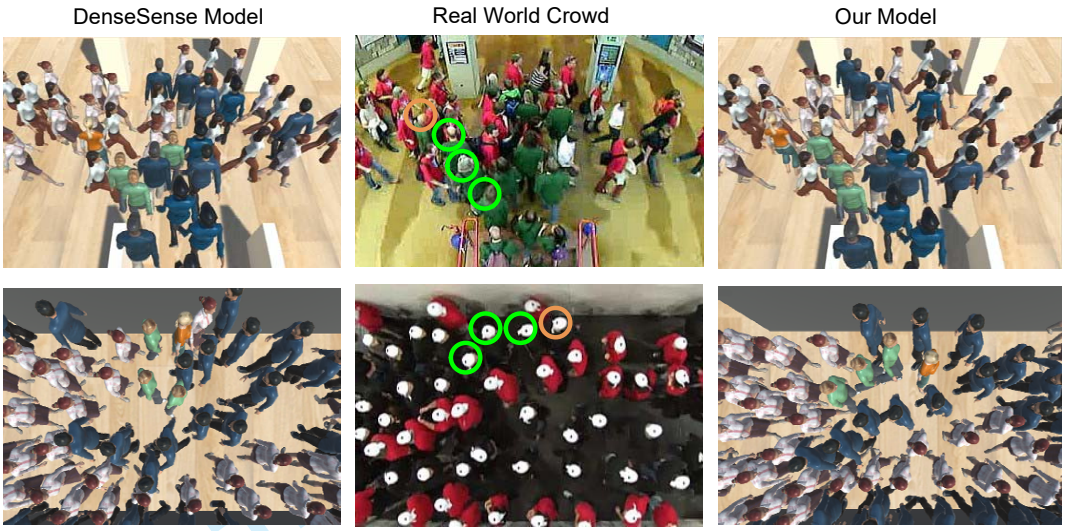

Figure 7: *Frames of real world crowd videos and 3D visualizations of simulation results from our model, ProactiveCrowd, and the DenseSense model for perpendicular (top) and bi-directional (bottom) crowd scenarios.*

, where Tr^{real} and Tr^{sim} are real world and simulated trajectories with size m and n respectively. $LCSS(T_r^{\text{real}}, T_r^{\text{sim}})$ is recursively defined as follows:

$$\begin{cases} 0 & \text{m=0 | n=0} \\ 1+LCSS_{\delta,\varepsilon}(Head(Tr^{\text{real}}), Head(Tr^{\text{sim}})) & d_E(a_n - b_m) < \varepsilon \\ & \& \ |n-m| \leq \delta \\ \max(LCSS_{\delta,\varepsilon}(Head(Tr^{\text{real}}), Tr^{\text{sim}}), & \\ LCSS_{\delta,\varepsilon}(Tr^{\text{real}}, Head(Tr^{\text{sim}}))) & \text{otherwise} \end{cases} \tag{20}$$

where δ is a user-defined constant which controls how far to stretch in time so as to match a given point from one trajectory to a point in another trajectory, ε is a user-defined constant specifying the matching threshold, $Head(Tr^{\text{real}})$ and $Head(Tr^{\text{sim}})$ are the first $n-1$ and $m-1$ points in Tr^{real} and Tr^{sim} respectively, $d_E(a_n - b_m)$ is the Euclidean distance between the nth point in Tr^{real} and mth point in Tr^{sim}. As pointed out in [VKG02], the *LCSS* provides an intuitive notion of similarity between trajectories by giving more weight to the similar portions of the sequences. It has been proven that the performance of *LCSS* is very stable and efficient for individual trajectory comparison [ZHT06].

Table 1: *Average S_{LCSS} and its standard deviation (in parentheses) over 10 independent runs, unit (%)*

Scenarios	DenseSense	Our model	p-value
Perpendicular	55.73(1.17)	78.20(2.5)	<0.0001
Bi-directional	75.12(1.76)	81.26(1.26)	<0.0005

Using the *LCSS* metric, we select and evaluate the agents performing gap seeking and following behaviors and some nearby agents which may be affected by such behaviors. For each selected agent, we extract its trajectory during the time period it (or its neighbouring agent) performs proactive steering behavior (usually about 5 to 10 seconds) and evaluate the S_{LCSS} with the corresponding trajectory data of real person. Then, we calculate the average of S_{LCSS} over all the selected agents. For the perpendicular crowd scenario, we randomly select and evaluate nineteen agents, which have performed proactive steering behaviors in our model or are in the vicinity of the agents performing proactive steering behaviors. Similarly, sixty of agents are selected and evaluated for the bi-directional crowd scenario. For comparison, we evaluate the same set of the agents using both our model, ProactiveCrowd, and the DenseSense model. For each model, we run 10 independent simulation runs with different random seeds. With the setting of $\delta = 20\%$ and $\varepsilon = 0.4$ for LCSS, the average S_{LCSS} over multiple runs and multiple agents for each scenario are shown in Table 1.

It can been seen that our model, ProactiveCrowd, can yield higher trajectory similarities for both scenarios than the DenseSense model. This indicates that our model can generate individual behaviors of agents that are more closer to real world data. The results are generally consistent with our observations from the visual inspection of 3D visualization. To test the statistical significance of our results, we conduct one-tailed two-sample t-test at 0.05 significance level. Our research hypothesis (i.e., the alternative hypothesis in t-test) is that the the simulation using our model can yield higher S_{LCSS} value compared to the one using the DenseSense model. The small p-values from t-tests as shown in Table 1 strongly suggest that our research hypothesis is supported.

From Table 1, we can also find that our model outperforms the DenseSense model more significantly in the perpendicular crowd scenario than in the bi-directional crowd scenario. This is probably due to that the gap seeking behaviors in the perpendicular crowd scenario are performed in a more abrupt manner as the environment is less constrained than the bi-directional scenario. Based on our analysis of the video data, the average turning angle for gap seek-

For Review Only

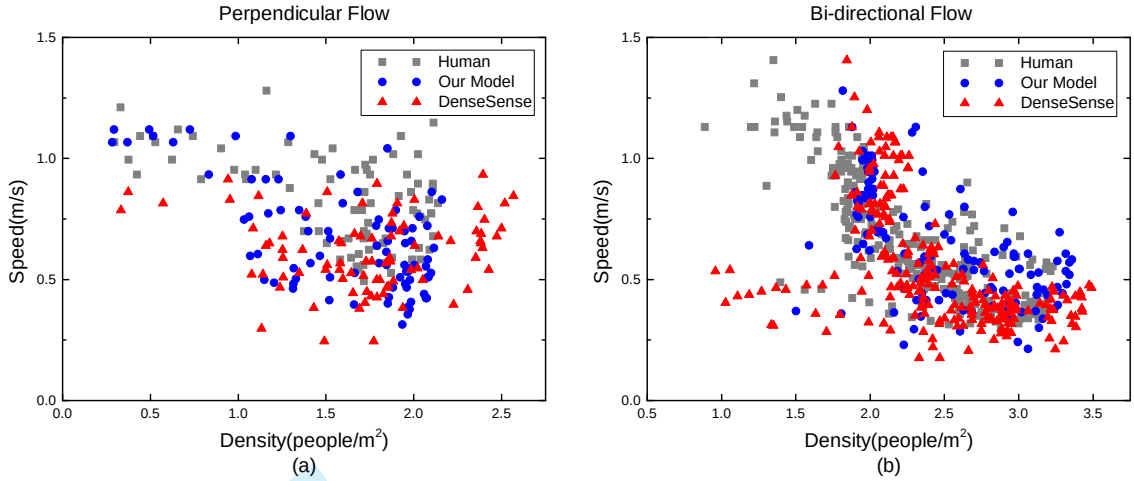

Figure 8: *Fundamental diagram for real human data, DenseSense and our model, ProactiveCrowd, in two scenarios.*

ing behavior is about 60 degree in the perpendicular crowd scenario whereas it is about 30 degree in the bi-directional. Thus, if such behavior is not properly simulated in the perpendicular crowd scenario, the trajectory difference to the real world data will be more apparent.

4.3.2. Crowd Level Evaluation

To evaluate the overall crowd dynamics, we compare the simulated crowd behavior with the behavior of real humans first by examining the speed-density relationship of agents using the fundamental diagram [WJGO*14], and second, by using a relative distance error metric proposed in [JHS07] which compares the temporal-spatial difference between the simulation results and real world data. We provide comparison results using our model, ProactiveCrowd, and the DenseSense model.

(a) *Fundamental diagrams evaluation*

The fundamental diagram is commonly used to evaluate pedestrian dynamics, which characterizes the observed relationship between pedestrian speed and density (usually speed decreases as density increases). The procedure to construct the fundamental diagram is similar to the one adopted in the work for the DenseSense model [NBCM15]. For each scenario, we first define a rectangular region where each agent must pass through. The regions are $3m \times 3m$ and $4m \times 3.6m$ rectangles at the center of the simulated area for the perpendicular and bi-directional crowd scenarios respectively. Then, we record the time interval for each agent to pass through the region and compute the average density of the agents in that region during that interval. In such a way, a single density-speed pair for each agent is recorded. The density-speed pairs for all agents are shown as a scatter plot in Figure 8.

In Figure 8, we present the fundamental diagrams extracted from the real human data, and the simulation results from both our model, ProactiveCrowd, and the DenseSense model. It can be observed that both models can generate the crowd dynamics that exhibit the downward trend of pedestrian's speed with the increase of crowd density. However, the distribution of density-speed pairs of our model appear to be more consistent with that of real world data. We can also observe some discrepancies between the results from the DenseSense model and real human data. For example, as shown in Figure 8 (a), the DenseSense model generates some agents with a high density (i.e., greater than $2.2 people/m^2$) in perpendicular crowd scenario, whereas we do not observe individuals with such density in the corresponding real world data. This indicates that the DenseSense model may generate congestions which did not happen in real-life situations. This is probably due to agent's inability to accurately perform the proactive steering behavior like gap seeking. In perpendicular crowd scenario, gap seeking behavior allows agent to redirect moving direction in a more abrupt manner, which can greatly help to avoid potential congestions.

(b) *Progressive distance error evaluation*

To further evaluate the spatial-temporal difference between the crowd dynamics generated by the simulation model and that obtained from real world data, we adopt the progressive distance error metric proposed in [JHS07], which is calculated as follows:

$$\sigma_{\text{err}} = \frac{1}{n * K} \sum_{i=1}^{n} \sum_{t_k=t_0}^{t_0+K*\Delta s} \frac{\|x_i^{\text{sim}}(t_k+T) - x_i^{\text{real}}(t_k+T)\|}{\|x_i^{\text{real}}(t_k+T) - x_i^{\text{real}}(t_k)\|} \qquad (21)$$

where $x_i^{\text{real}}(t_k)$ is the position of individual i in real world data at time t_k, T is a user-defined running period and $x_i^{\text{sim}}(t_k+T)$ is the position of corresponding agent i after it moves according to the simulation model for a running period of T with the simulation starts at time t_k. For a single agent, the evaluation are performed at different starting times t_k with a total number of K evaluations for every Δs time steps (15 time steps in our implementation). The overall distance error σ_{err} at crowd level is then obtained by averaging the error distances of all the simulated agents, where n is the total number of agents. Here, we use σ_{err} instead of direct *LCSS* metric, as such progressive difference metric has shown to be more effective to reflect the model difference for simulation spanning over a period of time with a larger number of agents [WJGO*14].

For Review Only

Table 2: *The relative distance error σ_{err} and its standard deviation (in parentheses) over 50 independent runs*

Scenarios	T	DenseSense	Our model	*p*-value
Perpendicular	1.675	0.37(0.021)	0.27(0.017)	<0.0001
	2.68	0.39(0.035)	0.22(0.029)	<0.0005
Bi-directional	1.56	0.57(0.025)	0.50(0.032)	<0.0001
	2.5	0.50(0.024)	0.42(0.015)	<0.0005

For model evaluation based on σ_{err}, we first perform the simulations using our model and the DenseSense model for 50 independent runs respectively and each run is set with different random seed. We then calculate σ_{err} over 50 runs in two cases for each scenario: T=1.675s (i.e., 25 simulation time steps), T=2.68s (i.e., 40 simulation time steps) for perpendicular crowd scenario and T=1.56s (i.e., 25 simulation time steps), T=2.5s (i.e., 40 simulation time steps) for bi-directional crowd scenario. Table 2 shows the results of our evaluation. It can been seen that our proposed model has yielded smaller σ_{err} value against real world data than the DenseSense model for both scenarios under different running periods. This suggests that our model can generally produce crowd behaviors closer to the real world data. It can also be seen in Table 2 that simulation results are better in perpendicular crowd scenario than bi-directional crowd scenario for both models. This is probably because that bi-directional scenario has a denser crowd within a constrained environment where more frequent local collision avoidances may occur and cause the actual velocities of agents be more easily deviated from the desired velocity for proactive steering.

To test the statistical significance of our results, we also conduct one-tailed two-sample *t*-test at 0.05 significance level. Our research hypothesis (i.e., the alternative hypothesis in *t*-test) is that the simulation using our proposed model can generate crowd behaviors which have smaller σ_{err} value compared to the one using the DenseSense model. The small *p*-values from *t*-tests as shown in Table 2 strongly suggest that our research hypothesis is supported.

5. Conclusions

In real-life scenarios, we often observe people adopt proactive steering strategies to avoid potential collisions with others. For agent-based crowd simulation, how to incorporate these proactive steering strategies is essential to ensure the realism of simulation. In this paper, we present ProactiveCrowd, a behavior-based approach to modeling proactive steering behaviors for agent-based crowd simulation. An overall modeling framework is introduced which determines how a suitable proactive steering behavior should be selected in a given situation and how to execute the selected behavior. With the focus on intersecting crowd scenario, the detailed behavior models for two inter-related proactive steering behaviors, namely *gap seeking* and *following*, are proposed. By applying our models to two real-world scenario, we demonstrated the validity of our models with the comparison of another state-of-the-art crowd model (i.e., the DenseSense model). Our ProactiveCrowd shows better or at least comparable performances in terms of the realism of simulation results. It also offers a more intuitive and direct way to capture the proactive steering behaviors of agents in contrast to most of the prediction-based models.

Currently, the two behaviors being modelled in ProactiveCrowd, *gap seeking* and *following*, are mainly observed in the scenarios where two crowds are intersecting. When ProactiveCrowd is applied to more complex scenarios (e.g., multiple flows of crowds in a shopping mall or train station), using just gap seeking and following may become inadequate. Some other proactive steering strategies, such as overtaking and stop-and-go, may be considered. Therefore, our future work is to investigate more crowd scenarios and propose new proactive steering behavior models to complement the existing ones. In addition, how different local collision avoidance models as well as path planning models could affect the performance of our models also needs to be further investigated.

References

[BNCM14] BEST A., NARANG S., CURTIS S., MANOCHA D.: DenseSense: Interactive crowd simulation using density-dependent filters. In *Symposium on Computer Animation* (2014), pp. 97–102. 3, 7

[BP15] BRUNEAU J., PETTRÉ J.: Energy-efficient mid-term strategies for collision avoidance in crowd simulation. In *Proceedings of the 14th ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (2015), ACM, pp. 119–127. 1, 3

[DP85] DECHTER R., PEARL J.: Generalized best-first search strategies and the optimality of A*. *Journal of the ACM (JACM) 32*, 3 (1985), 505–536. 2

[FETA16] FANG J., EL-TAWIL S., AGUIRRE B.: Leader–follower model for agent based simulation of social collective behavior during egress. *Safety Science 83* (2016), 40–47. 6

[GNCL14] GOLAS A., NARAIN R., CURTIS S., LIN M. C.: Hybrid long-range collision avoidance for crowd simulation. *IEEE transactions on visualization and computer graphics 20*, 7 (2014), 1022–1034. 1, 3

[Her] Hermes project:. http://www.asim.uni-wuppertal.de/en/research/hermes.html. 2, 7, 8

[HFV00] HELBING D., FARKAS I., VICSEK T.: Simulating dynamical features of escape panic. *Nature 407*, 6803 (2000), 487–490. 3, 8

[HLPW14] HOU L., LIU J.-G., PAN X., WANG B.-H.: A social force evacuation model with the leadership effect. *Physica A: Statistical Mechanics and its Applications 400* (2014), 93–99. 6, 7

[HZW*10] HU N., ZHOU S., WU Z., ZHOU M., CHO B. E. K.: Spatial-temporal patterns and pedestrian simulation. *Computer Animation and Virtual Worlds 21*, 3-4 (2010), 387–399. 3

[JHS07] JOHANSSON A., HELBING D., SHUKLA P.: Specification of the social force pedestrian model by evolutionary adjustment to video tracking data. *Advances in complex systems 10*, supp02 (2007), 271–288. 8, 10

[KHvBO09] KARAMOUZAS I., HEIL P., VAN BEEK P., OVERMARS M. H.: A predictive collision avoidance model for pedestrian simulation. In *Motion in Games*. Springer, 2009, pp. 41–52. 3

[KSHF09] KAPADIA M., SINGH S., HEWLETT W., FALOUTSOS P.: Egocentric affordance fields in pedestrian steering. In *Proceedings of the 2009 symposium on Interactive 3D graphics and games* (2009), ACM, pp. 215–223. 3

[LCZM16] LUO L., CHAI C., ZHOU S., MA J.: Modeling gap seeking behaviors for agent-based crowd simulation. In *Proceedings of the 29th International Conference on Computer Animation and Social Agents* (2016), ACM, pp. 37–43. 2

[LJK*12] LEMERCIER S., JELIC A., KULPA R., HUA J., FEHRENBACH J., DEGOND P., APPERT-ROLLAND C., DONIKIAN S., PETTRÉ J.: Realistic following behaviors for crowd simulation. In *Computer Graphics Forum* (2012), vol. 31, Wiley Online Library, pp. 489–498. 6

For Review Only

[NBCM15] NARANG S., BEST A., CURTIS S., MANOCHA D.: Generating pedestrian trajectories consistent with the fundamental diagram based on physiological and psychological factors. *PLoS one 10*, 4 (2015), e0117856. 3, 10

[OH15] OZCAN C. Y., HACIOMEROGLU M.: A path-based multi-agent navigation model. *The Visual Computer* (2015), 1–10. 3

[OPOD10] ONDŘEJ J., PETTRÉ J., OLIVIER A.-H., DONIKIAN S.: A synthetic-vision based steering approach for crowd simulation. In *ACM Transactions on Graphics (TOG)* (2010), vol. 29, ACM, p. 123. 1

[PAB07] PELECHANO N., ALLBECK J., BADLER N.: Controlling individual agents in high-density crowd simulation. In *Proceedings of the 2007 ACM SIGGRAPH/Eurographics symposium on Computer animation* (San Diego, USA, 2007), pp. 99–108. 3

[PB06] PELECHANO N., BADLER N. I.: Modeling crowd and trained leader behavior during building evacuation. *IEEE Computer Graphics and Applications 26*, 6 (2006), 80–86. 3

[PCBS11] PLAUE M., CHEN M., BÄRWOLFF G., SCHWANDT H.: Trajectory extraction and density analysis of intersecting pedestrian flows from video recordings. In *Photogrammetric Image Analysis*. Springer, 2011, pp. 285–296. 8

[PPD07] PARIS S., PETTRÉ J., DONIKIAN S.: Pedestrian reactive navigation for crowd simulation: a predictive approach. In *Computer Graphics Forum* (2007), vol. 26, Wiley Online Library, pp. 665–674. 1, 3

[PSAB08] PELECHANO N., STOCKER C., ALLBECK J., BADLER N.: Being a part of the crowd: towards validating vr crowds using presence. In *Proceedings of the 7th international joint conference on Autonomous agents and multiagent systems* (Estoril, Portugal, 2008), pp. 136–142. 1

[PVdBC*11] PATIL S., VAN DEN BERG J., CURTIS S., LIN M., MANOCHA D.: Directing crowd simulations using navigation fields. *IEEE Transactions on Visualization and Computer Graphics 17*, 2 (2011), 244–254. 3

[SGA*07] SUD A., GAYLE R., ANDERSEN E., GUY S., LIN M., MANOCHA D.: Real-time navigation of independent agents using adaptive roadmaps. In *Proceedings of the 2007 ACM symposium on Virtual reality software and technology* (2007), ACM, pp. 99–106. 3

[SMD] Simulation of Multi Destination Pedestrain Crowds (SMDPC) project:. http://www.math.tu-berlin.de/projekte/smdpc/. 2, 7, 8

[Sno00] SNOOK G.: Simplified 3d movement and pathfinding using navigation meshes. *Game Programming Gems 1*, 1 (2000), 288–304. 3

[ST05] SHAO W., TERZOPOULOS D.: Autonomous pedestrians. In *Proceedings of the 2005 ACM SIGGRAPH/Eurographics symposium on Computer animation* (Los Angeles, USA, 2005), pp. 19–28. 1, 2

[TFB*11] TSAI J., FRIDMAN N., BOWRING E., BROWN M., EPSTEIN S., KAMINKA G., MARSELLA S., OGDEN A., RIKA I., SHEEL A., ET AL.: Escapes: evacuation simulation with children, authorities, parents, emotions, and social comparison. In *Proceedings of the 10th International Conference on Autonomous Agents and Multiagent Systems* (Taipei, Taiwan, 2011), pp. 457–464. 1

[Tha07] THALMANN D.: *Crowd simulation*. Wiley Online Library, 2007. 1

[VDBGLM11] VAN DEN BERG J., GUY S. J., LIN M., MANOCHA D.: Reciprocal n-body collision avoidance. In *Robotics research*. Springer, 2011, pp. 3–19. 1, 3

[VKG02] VLACHOS M., KOLLIOS G., GUNOPULOS D.: Discovering similar multidimensional trajectories. In *Data Engineering, 2002. Proceedings. 18th International Conference on* (2002), IEEE, pp. 673–684. 8, 9

[WJGO*14] WOLINSKI D., J GUY S., OLIVIER A.-H., LIN M., MANOCHA D., PETTRÉ J.: Parameter estimation and comparative evaluation of crowd simulations. In *Computer Graphics Forum* (2014), vol. 33, Wiley Online Library, pp. 303–312. 10

[ZCC*10] ZHOU S., CHEN D., CAI W., LUO L., LOW M., TIAN F., TAY V.-H., ONG D., HAMILTON B.: Crowd modeling and simulation technologies. *ACM Transactions on Modeling and Computer Simulation (TOMACS) 20*, 4 (2010), 20. 1

[ZCLL14] ZHONG J., CAI W., LUO L., LEES M.: Ea-based evacuation planning using agent-based crowd simulation. In *Proceedings of the 2014 Winter Simulation Conference* (Savannah, USA, 2014), pp. 395–406. 1

[ZHT06] ZHANG Z., HUANG K., TAN T.: Comparison of similarity measures for trajectory clustering in outdoor surveillance scenes. In *18th International Conference on Pattern Recognition (ICPR'06)* (2006), vol. 3, IEEE, pp. 1135–1138. 9

[ZKSS12] ZHANG J., KLINGSCH W., SCHADSCHNEIDER A., SEYFRIED A.: Ordering in bidirectional pedestrian flows and its influence on the fundamental diagram. *Journal of Statistical Mechanics: Theory and Experiment 2012*, 02 (2012), P02002. 8

[ZZC16] ZHAO M., ZHONG J., CAI W.: A role-dependent data-driven approach for high density crowd behavior modeling. In *Proceedings of the 2016 annual ACM Conference on SIGSIM Principles of Advanced Discrete Simulation* (2016), ACM, pp. 89–97. 3

For Review Only